In the Claims

1. (Currently Amended) A method for memory management in smart card controllers or similar restricted hardware environment by writing of data into a data space in a persistent memory, said method comprising ~~steps of~~:

a) splitting the persistent memory into blocks with fixed data length having logical block numbers (LBN);

b) selecting the size of blocks as such that it is equal ~~to or~~to, or equivalent to an integer ratio ~~of the~~of, the length of a page in EEPROM to the physical size of the pages of the EEPROM memory existing on the card;

c) providing a Block Allocation Table (BAT) in order to calculate the physical place of the block in memory from the logical block number;

d) defining a bit existing in each block header, whereby this bit corresponds to a bit existing in a commit block;

e) where toggling of a bit in the commit block toggles the validity of the corresponding memory block;

f) replacing individual memory blocks by each other to accomplish a secure write mechanism by:

1) writing the update data for a block together with the unchanged data of the block to a new formerly free block;

2) committing the operation by writing a new commit field after finishing the update process; and

3) erasing the old data blocks and updating the BAT so that the physical blocks for the concerned logical blocks are exchanged;

g) typically all commit bits of the field are located in one EEPROM page (a commit block) to prevent the system from losing a valid commit field (respectively commit block) if a power failure occurs during updating the commit block, the commit clock is doubled and only one of the two commit blocks is valid at a time whereby an update of the commit block is always done by writing to the block not written to at the last update, because this is the invalid commit block whereby the validity from the

invalid block is determined by a two-bit counter, which is added to each commit block (C0, C1).

2. (Original) The method according to claim 1, including the step of splitting a whole block into individual segments, whereby each fragment is belonging to a different data object.

3. (Original) The method according to claim 2, including the step of identifying a corresponding segment through the block number of the whole block and the number of the individual segment.

4. (Original) The method according to claim 2, including defining a block header in the block with a list of entries providing information to localize the segments as well as defining their length.

5. (Original) The method according to claim 1, wherein a linkage between blocks by writing the LBN of the following block to the header of the leading block is provided.

6. (Cancelled)

7. (Original) The method according to claim 1, wherein some kinds of blocks are organized in form of a ring list.

8. (Cancelled)

9. (Cancelled)

10. (Currently Amended) A device with a persistent memory and a block structure comprising:

a) a memory managing system using a block oriented memory structure;

b) blocks with the same length and identifying them by their logical block number (LBN);

c) a ~~block allocation table~~Block Allocation Table (BAT) to resolve the logical block number to a physical block number (PBN) and its physical address; and

d) a linkage between blocks by writing the LBN of the following block to the header of the leading one;

11. (Cancelled)

12. (Currently Amended) ~~A~~ The device according to claim 10 characterized by blocks with fixed block size, which can contain several independent memory segments, belonging to different logical data units.

13. (Currently Amended) ~~A~~ The device according to claim 10 characterized in that the BAT is held in persistent memory (EEPROM).

14. (Currently Amended) ~~A~~ The device according to claim 10 characterized in that the BAT is held in non-persistent memory ~~(RAM BAT)~~ and re-initialized on startup, whereby the structure of a memory block consists of a block header and a data area and the header consists of status data containing MSB logical block number of the own block and MSB of the next block number.

15. (New) A method for memory management in smart card controllers or similar restricted hardware environment by writing of data into a data space in a persistent memory, said method comprising:

a) splitting the persistent memory into blocks with fixed data length having logical block numbers (LBN);

b) selecting the size of blocks as such that it is equal to, or equivalent to an integer ratio of, the length of a page in EEPROM to the physical size of the pages of the EEPROM memory existing on the card;

c) providing a Block Allocation Table (BAT) in order to calculate the physical place of the block in memory from the logical block number;

d) defining a bit existing in each block header, whereby this bit corresponds to a bit existing in a commit block;

e) where toggling of a bit in the commit block toggles the validity of the corresponding memory block.

16. (New) The method according to claim 15, wherein the commit bits are managed on a physical level.

17. (New) The method according to claim 1, wherein the commit bits are managed on a physical level.